

Scene Grammars, Factor Graphs, and Belief Propagation

Jeroen Chua
 Department of Computer Science
 Brown University
 jeroen_chua@brown.edu,

Pedro F. Felzenszwalb
 School of Engineering and Department of Computer Science
 Brown University
 pff@brown.edu

June 7, 2016

Abstract

We consider a class of probabilistic grammars for generating scenes with multiple objects. Probabilistic scene grammars capture relationships between objects using compositional rules that provide important contextual cues for inference with ambiguous data. We show how to represent the distribution defined by a probabilistic scene grammar using a factor graph. We also show how to efficiently perform message passing in this factor graph. This leads to an efficient approach for inference with a grammar model using belief propagation as the underlying computational engine. Inference with belief propagation naturally combines bottom-up and top-down contextual information and leads to a robust algorithm for aggregating evidence. We show experiments on two different applications to demonstrate the generality of the framework. The first application involves detecting curves in noisy images, and we address this problem using a grammar that generates a collection of curves using a first-order Markov process. The second application involves localizing faces and parts of faces in images. In this case, we use a grammar that captures spatial relationships between the parts of a face. In both applications the same framework leads to robust inference algorithms that can effectively combine weak local information to reason about a scene.

1 Introduction

The primary motivation of this work is that objects and scenes can be represented using hierarchical structures defined by compositional rules. For instance, faces are composed of eyes, nose, mouth. Similarly, geometric objects such as curves can be defined in terms of shorter curves that are recursively described. A hierarchical structure defined by compositional rules defines a rich description of a scene that captures both the presence of different objects and relationships among them. Moreover, compositional rules provide contextual cues for inference with ambiguous data. For example, the presence of some parts of a face in a scene provides contextual cues for the presence of other parts.

In the models we consider, every object has a type from a finite alphabet and a pose from a finite but large pose space. While classical language models generate sentences using a single derivation, the grammars we consider generate scenes using multiple derivations. These derivations can be unrelated or they can share sub-derivations. This allows for very general descriptions of scenes.

We show how to represent the distributions defined by probabilistic scene grammars using factor graphs, and we use loopy belief propagation (LBP) [17, 15] for approximate inference. Inference with LBP simultaneously combines “bottom-up” and “top-down” contextual information. For example, when faces are defined using a composition of eyes, nose and mouth, the evidence for a face or one of its parts provides contextual influence for the whole composition. Inference via message passing naturally captures chains of contextual evidence. LBP also naturally combines multiple contextual cues. For example, the presence of an eye may provide contextual evidence for a face at two different

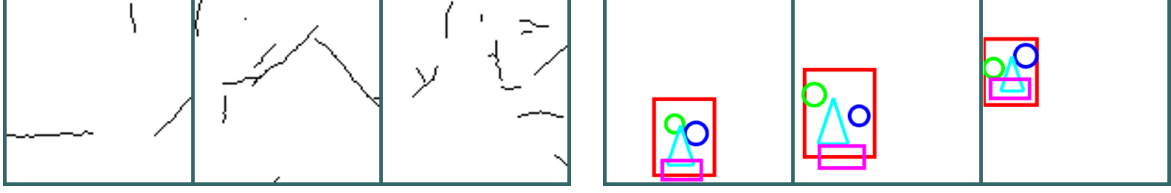


Figure 1: Left: Samples of contour maps generated by our grammar model of curves. Right: Samples of scenes with a single face generated by our grammar model of faces.

locations because a face has a left and a right eye. However, the presence of two eyes side by side provides strong evidence for a single face between them.

We demonstrate the practical feasibility of the approach on two very different applications: curve detection and face localization. Figure 1 shows samples from the two different grammars we use for the experimental results. The contributions of our work include (1) a unified framework for contextual modeling that can be used in a variety of applications; (2) a construction that maps a probabilistic scene grammar to a factor graph together with an efficient message passing scheme; and (3) experimental results showing the effectiveness of the approach.

Probabilistic grammars and compositional models are widely used for parsing sentences in natural language processing [16]. Recursive descriptions of objects using grammars and production rules have also been widely used in computer graphics to generate geometric objects, biological forms, and landscapes [19]. A variety of other compositional models have been used in computer vision. The models we consider are closely related to the Markov backbone model in [14]. Other related approaches include [2, 20, 8, 21, 12, 10]. These previous methods have relied on MCMC or heuristic methods for inference, or dynamic programming for scenes with single objects. The models we consider generalize part-based models for object detection such as pictorial structures [11, 7] and constellations of features [3]. In particular, the grammars we consider define objects that are composed of parts but allow for modeling objects with variable structure. The models we consider also explicitly capture scenes with multiple objects.

2 Probabilistic Scene Grammars and a Factor Graph Representation

Our point of departure is a probabilistic scene grammar that defines a distribution over scenes. The approach is based on the Markov backbone from [14]. Scenes are defined using a library of building blocks, or *bricks*, that have a type and a pose. Bricks are generated spontaneously or through expansions of other bricks. This leads to a hierarchical organization of the elements of a scene.

Definition 2.1. A probabilistic scene grammar \mathcal{G} consists of

1. A finite set of symbols, or types, Σ .
2. A finite pose space, Ω_A , for each symbol $A \in \Sigma$.
3. A finite set of production rules, \mathcal{R} . Each rule $r \in \mathcal{R}$ is of the form $A_0 \rightarrow \{A_1, \dots, A_{N_r}\}$, where $A_i \in \Sigma$. We use \mathcal{R}_A to denote the rules with symbol A in the left-hand-side (LHS). We use $A_{r,i}$ to denote the i -th symbol in the right-hand-side (RHS) of a rule r .
4. Rule selection probabilities, $P(r)$, with $\sum_{r \in \mathcal{R}_A} P(r) = 1$ for each symbol $A \in \Sigma$.
5. For each rule $r = A_0 \rightarrow \{A_1, \dots, A_{N_r}\}$ we have categorical distributions $g_{r,i}(z|\omega)$ defining the probability of a pose z for A_i conditional on a pose ω for A_0 .
6. Self-rooting probabilities, ϵ_A , for each symbol $A \in \Sigma$.
7. A noisy-or parameter, ρ .

The bricks defined by \mathcal{G} are pairs of symbols and poses,

$$\mathcal{B} = \{(A, \omega) \mid A \in \Sigma, \omega \in \Omega_A\}.$$

Definition 2.2. A scene S is defined by

1. A set $\mathcal{O} \subseteq \mathcal{B}$ of bricks that are present in S .
2. A rule $r \in \mathcal{R}_A$ for each brick $(A, \omega) \in \mathcal{O}$, and a pose $z \in \Omega_{A_i}$ for each A_i in the RHS of r .

Let $H = (\mathcal{B}, E)$ be a directed graph capturing which bricks can generate other bricks in one production. For each rule r , if $g_{r,i}(z|\omega) > 0$, we include $((A_0, \omega), (A_i, z))$ in E . We say a grammar \mathcal{G} is *acyclic* if H is acyclic.

A *topological ordering* of \mathcal{B} is an ordering of the bricks such that (A, ω) appears before (B, z) whenever (A, ω) can generate (B, z) . When \mathcal{G} is acyclic we can compute a topological ordering of \mathcal{B} by topological sorting the vertices of H .

Definition 2.3. An acyclic grammar defines a distribution over scenes, $P(S)$, through the following generative process.

1. Initially $\mathcal{O} = \emptyset$.
2. For each brick $(A, \omega) \in \mathcal{B}$ we add (A, ω) to \mathcal{O} independently with probability ϵ_A .
3. We consider the bricks in \mathcal{B} in a topological ordering. When considering (A, ω) , if $(A, \omega) \in \mathcal{O}$ we expand it.
4. To expand (A, ω) we select a rule $r \in \mathcal{R}_A$ according to $P(r)$ and for each A_i in the RHS of r we select a pose z according to $g_{r,i}(z|\omega)$. We add (A_i, z) to \mathcal{O} with probability ρ .

Note that because of the topological ordering of the bricks, no brick is included in \mathcal{O} after it has been considered for expansion. In particular each brick in \mathcal{O} is expanded exactly once. This leads to derivation trees rooted at each brick in the scene. The expansion of two different bricks can generate the same brick, and this leads to a “collision” of derivations. When two derivations collide they share a sub-derivation rooted at the point of collision. Derivations terminate using rules of the form $A \rightarrow \emptyset$, or through early termination of a branch with probability ρ .

2.1 Factor Graph Representation

We can represent the distribution over scenes, $P(S)$, using a factor graph with binary variables.

Definition 2.4. A scene S generated by an acyclic grammar \mathcal{G} defines a set of random variables,

$$X(A, \omega) \in \{0, 1\} \quad \forall (A, \omega) \in \mathcal{B} \quad (1)$$

$$R(A, \omega, r) \in \{0, 1\} \quad \forall (A, \omega) \in \mathcal{B}, \forall r \in \mathcal{R}_A \quad (2)$$

$$G(A, \omega, r, i, z) \in \{0, 1\} \quad \forall (A, \omega) \in \mathcal{B}, \forall r \in \mathcal{R}_A, 1 \leq i \leq N_r, \forall z \in \Omega_{A_{r,i}} \quad (3)$$

where

1. $X(A, \omega) = 1$ if $(A, \omega) \in \mathcal{O}$.
2. $R(A, \omega, r) = 1$ if rule r is used to expand (A, ω) .
3. $G(A, \omega, r, i, z) = 1$ when (A, ω) is expanded with rule r , and z is the pose selected for A_i .

Let $G(A, \omega)$ be the vector of variables $G(B, z, r, i, \omega)$ where $A_{r,i} = A$. We have $X(A, \omega) = 0$ when $X(A, \omega)$ is not generated spontaneously or by the expansion of another other brick. Therefore,

$$P(X(A, \omega)|G(A, \omega)) = \begin{cases} (1 - \rho)^c (1 - \epsilon_A) & X(A, \omega) = 0 \\ 1 - (1 - \rho)^c (1 - \epsilon_A) & \text{otherwise} \end{cases} \quad (4)$$

where c is the number of variables in $G(A, \omega)$ with value 1.

Let $R(A, \omega)$ be the vector of random variables $R(A, \omega, r)$. The generative process determines $R(A, \omega)$ by selecting a rule $r \in \mathcal{R}_A$ for expanding (A, ω) when $X(A, \omega) = 1$, and no rule is selected when $X(A, \omega) = 0$. Therefore,

$$P(R(A, \omega)|X(A, \omega)) = \begin{cases} 1 & R(A, \omega) = 0, X(A, \omega) = 0 \\ P(r) & R(A, \omega) = I(r), X(A, \omega) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where $I(r)$ is an indicator vector for $r \in \mathcal{R}_A$

Let $G(A, \omega, r, i)$ be the vector of random variables $G(A, \omega, r, i, z)$. The generative process selects a pose z for $A_{r,i}$ if the rule r is used to expand a brick. Therefore,

$$P(G(A, \omega, r, i)|R(A, \omega, r)) = \begin{cases} 1 & G(A, \omega, r, i) = 0, R(A, \omega, r) = 0 \\ g_{r,i}(z|\omega) & G(A, \omega, r, i) = I(z), R(A, \omega, r) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where $I(z)$ is an indicator vector for $z \in \Omega_{A_{r,i}}$.

The joint distribution, $P(X, R, G)$, defined by an acyclic grammar can be expressed using a factored representation following the structure of the generative process defined by \mathcal{G} ,

$$P(X, R, G) = \prod_{(A, \omega) \in \mathcal{B}} \left(P(X(A, \omega)|G(A, \omega)) P(R(A, \omega)|X(A, \omega)) \prod_{r \in \mathcal{R}_A, 1 \leq i \leq N_r} P(G(A, \omega, r, i)|R(A, \omega, r)) \right). \quad (7)$$

We can express $P(X, R, G)$ using a factor graph over the binary variables, with a factor for each term in the product above. The factors in the factor graph representation are

$$\Psi_{(A, \omega)}^1(X(A, \omega), G(A, \omega)) = P(X(A, \omega)|G(A, \omega)) \quad (8)$$

$$\Psi_{(A, \omega)}^2(R(A, \omega), X(A, \omega)) = P(R(A, \omega)|X(A, \omega)) \quad (9)$$

$$\Psi_{(A, \omega, r, i)}^3(G(A, \omega, r, i), R(A, \omega, r)) = P(G(A, \omega, r, i)|R(A, \omega, r)). \quad (10)$$

Although we have assumed an acyclic grammar in the derivation of the distribution $P(X, R, G)$ in equation (7), the factor graph construction can also be applied to arbitrary grammars. This makes it possible to define probability distributions over scenes using cyclic grammars, without relying on the generative process formulation.

3 Inference Using Belief Propagation

To perform approximate inference with the factor graph representation, we use loopy belief propagation (LBP) [17, 15]. Here we describe how to compute LBP messages efficiently for the factor graphs that represent scene grammars.

The factors in our model are of one of two kinds: The factor Ψ_1 defined in equation (8) captures a noisy-OR distribution, and the factors Ψ_2 and Ψ_3 defined in equations (9) and (10) capture categorical distributions in which the outcome probabilities depend on the state of a switching random variable. Figure 2 shows the local graphical representation for the two types of factors. The computation of messages from variables to factors follows the standard LBP equations. Below we describe how to efficiently compute the messages from factors to variables. The computational complexity of message updates for both kinds of factors is *linear* in the degree of the factor. In the derivations below we assume all messages have non-zero value.

3.1 Message passing for noisy-OR factors

Consider a factor $F(y_1, \dots, y_N, z)$ that represents a noisy-OR relationship between binary inputs y_1, \dots, y_N , and a binary output z . Suppose we have a leak in the noisy-OR with probability ϵ and independent failure parameter $1 - \rho$.



Figure 2: The two types of factors in our model: noisy-OR and categorical.

We define $\beta = -\log(1 - \rho)$. We can write the factor F as

$$F(y_1, \dots, y_N, 0) = (1 - \epsilon) \prod_{i=1}^N \exp(-\beta y_i), \quad (11)$$

$$F(y_1, \dots, y_N, 1) = 1 - F(y_1, \dots, y_N, 0). \quad (12)$$

The message passing equations are straightforward to derive and we simply state them here,

$$1 - \mu_{F \rightarrow Z}(1) = \mu_{F \rightarrow Z}(0) = (1 - \epsilon) \prod_{i=1}^N (\mu_{Y_i \rightarrow F}(0) + \mu_{Y_i \rightarrow F}(1) \exp(-\beta)) \quad (13)$$

$$r_i = (1 - \epsilon) \prod_{j \neq i} (\mu_{Y_j \rightarrow F}(0) + \mu_{Y_j \rightarrow F}(1) \exp(-\beta)) \quad (14)$$

$$\mu_{F \rightarrow Y_i}(y_i) = r_i \exp(-\beta y_i) (\mu_{Z \rightarrow F}(0) - \mu_{Z \rightarrow F}(1)) + \mu_{Z \rightarrow F}(1). \quad (15)$$

3.2 Message passing for categorical factors

Consider a factor $F(y, z_1, \dots, z_N)$ that represents a mixture of categorical distributions. The binary values z_1, \dots, z_N specify the outcome and y controls the outcome probabilities. Concretely,

$$F(y, z_1, \dots, z_N) = \begin{cases} 0 & \sum_{i=1}^N z_i \neq 1 \\ \prod_{i=1}^N (\theta_y^i)^{z_i} & \text{otherwise} \end{cases} \quad (16)$$

where θ_y^i is the probability of the i -th outcome with the mixture component defined by y .

In this case we can derive the following message passing equations,

$$\mu_{F \rightarrow Z_i}(1) = \left(\prod_{j \neq i} \mu_{Z_j \rightarrow F}(0) \right) \sum_y \mu_{Y \rightarrow F}(y) \theta_y^i \quad (17)$$

$$\mu_{F \rightarrow Z_i}(0) = \left(\prod_{j \neq i} \mu_{Z_j \rightarrow F}(0) \right) \sum_y \mu_{Y \rightarrow F}(y) \left(\sum_{j \neq i} \frac{\mu_{Z_j \rightarrow F}(1)}{\mu_{Z_j \rightarrow F}(0)} \theta_y^j \right) \quad (18)$$

$$\mu_{F \rightarrow Y}(y) = \left(\prod_{i=1}^N \mu_{Z_i \rightarrow F}(0) \right) \sum_{i=1}^N \frac{\mu_{Z_i \rightarrow F}(1)}{\mu_{Z_i \rightarrow F}(0)} \theta_y^i. \quad (19)$$

4 Learning Model Parameters

For a grammar with fixed structure we can use EM to learn the the production rule probabilities, $P(r)$, and the self-rooting parameters, ϵ_A . The approach involves iterative updates. In each iteration, we (1) use LBP to compute (approximate) conditional marginal probabilities on training examples with the current model parameters, and (2) update the model parameters according to sufficient statistics derived from the output of LBP.

Let $Q_e(A, \omega, r)$ be the marginal probability of brick (A, ω) being expanded using rule r in the training example e . In the factor graph representation, this corresponds to the marginal probability that a random variable takes a particular value, $P(R(A, \omega, r) = 1)$, a quantity that is approximated by the output of LBP. The update for $P(r)$ is,

$$P(r) = \frac{1}{Z_A} \sum_e \sum_{\omega \in \Omega_a} Q_e(A, \omega, r). \quad (20)$$

The value of Z_A is determined by normalizing probabilities over $r \in \mathcal{R}_A$. We update the self-rooting parameters, ϵ_A , in an analogous way, using approximate marginals computed by LBP.

5 Experiments

To demonstrate the generality of our approach we conducted experiments with two different applications: curve detection, and face localization. Previous approaches for these problems typically use fairly distinct methods. Here, we demonstrate we can handle both problems within the same framework. In particular we have used a single implementation of a general computational engine for both applications. The computational engine can perform inference and learning using arbitrary scene grammars. We report the speed of inference as performed on a laptop with an Intel[®] i7 2.5GHz CPU and 16 GB of RAM. Our framework is implemented in Matlab/C using a single thread.

5.1 Curve detection

To study curve detection we used the Berkeley Segmentation Dataset (BSD500) [1] following the experimental setup described in [9]. The dataset contains natural images and object boundaries manually marked by human annotators. For our experiments, we used the standard split of the dataset with 200 training images and 200 test images. For each image we use the boundaries marked by a single human annotator to define ground-truth binary contour maps J .

From a binary contour map J we generate a noisy image I by sampling each pixel $I(x, y)$ independently from a normal distribution whose mean depends on the value of $J(x, y)$.

$$I(x, y) \sim N(\mu(J(x, y)), \sigma). \quad (21)$$

For our experiments, we used $\mu(0) = 150$, $\mu(1) = 100$, $\sigma = 40$.

To model binary contour maps we use a first-order Markov process that generates curves of different orientations and varying lengths. The grammar is defined by two symbols: C (oriented curve element) and J (curve pixel). We consider curves in one of 8 possible orientations. For an image of size $[n, m]$, the pose space for C is an $(n \times m) \times 8$ grid and the pose space for J is an $n \times m$ grid.

We can express the rules of the grammar as

$$C((x, y), \theta) \rightarrow J(x, y) \quad 0.05 \quad (22)$$

$$C((x, y), \theta) \rightarrow J(x, y), C((x, y) + R_\theta(1, 0), \theta) \quad 0.73 \quad (23)$$

$$C((x, y), \theta) \rightarrow J(x, y), C((x, y) + R_\theta(1, +1), \theta) \quad 0.11 \quad (24)$$

$$C((x, y), \theta) \rightarrow J(x, y), C((x, y) + R_\theta(1, -1), \theta) \quad 0.11 \quad (25)$$

where $R_\theta(x, y)$ denotes a rotation of (x, y) by θ . Consider generating a “horizontal” curve, with orientation $\theta = 0$, starting at pixel (x, y) . The process starts at the brick $C((x, y), 0)$. Expansion of this brick will generate a brick $J(x, y)$ to denote that pixel (x, y) is part of a curve in the scene. Expansion of $C((x, y), 0)$ with the first rule ends the curve, while expansion with one of the other rules continues the curve in one of the three pixels to the right of (x, y) .

The values on the right of the rules above indicate their probabilities. To learn the rule probabilities and self-rooting parameters, we used the approach outlined in Section 4. We show random contour maps J generated by this grammar in Figure 1. The model generates multiple curves in a single image due to the self-rooting parameters.

In Figure 3 we show curve detection results using the curve grammar for some examples from the BSDS500 test set (see the supplementary material for more results). We illustrate the estimated probability that each pixel is part of

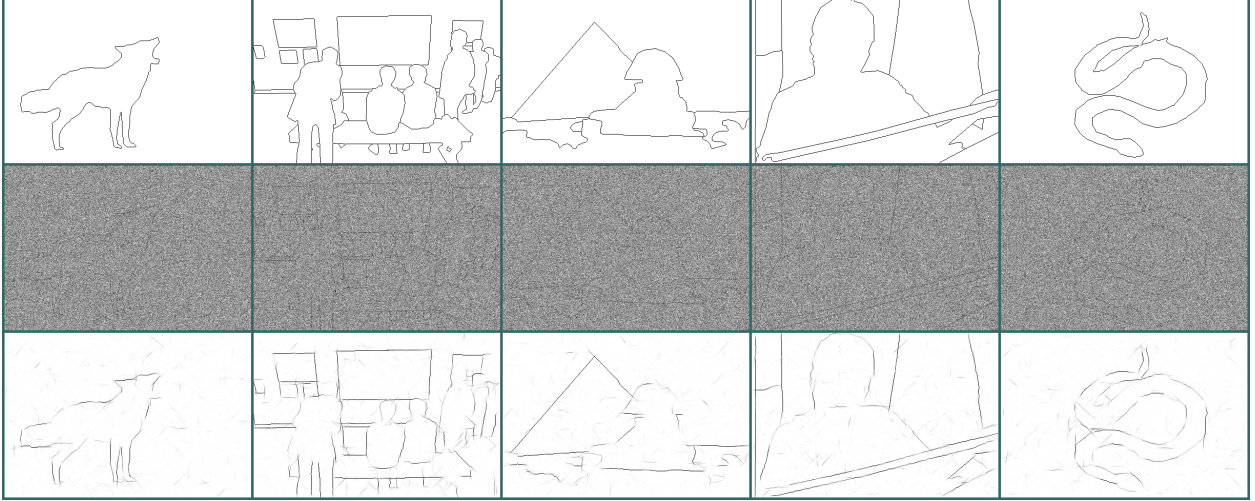


Figure 3: Curve detection results in the BSDS500 test set. Top row: Ground-truth contour maps. Middle row: Noisy observations, I . Bottom row: Estimated probability that a curve goes through each pixel, with dark values for high-probability pixels.

a curve, $P(X(J, (x, y)) = 1|I)$. This involves running LBP in the factor graph representing the curve grammar. For inference with an observed image, I , we use the model in equation (21). In the factor graph, this means the variable $X(J, (x, y))$ is connected to and receives a fixed-message from $I(x, y)$. Inference on a (481×321) test image took 1.5 hours.

For a quantitative evaluation we compute an AUC score, corresponding to the area under the precision-recall curve obtained by thresholding $P(X(J, (x, y)) = 1|I)$. We also evaluate a baseline “no-context” model, where the probability that a pixel belongs to a curve is computed using only the observation at that pixel. The grammar model obtained an AUC score 0.71 while the no-context baseline achieved an AUC score of 0.11. For comparison, in [9] an AUC score of 0.73 was reported for the single-scale Field-of-Patterns (FOP) model.¹

The use of contextual information defined by the curve grammar described here significantly improves the curve detection performance. Although our method performed well in detecting curves in extremely noisy images, the model has some trouble finding curves with high curvature. We believe this is primarily because the grammar we used does not have a notion of curvature. It is possible to define more detailed models of curves to improve performance. However, we note that a simple first-order model of curves with no curvature information is sufficient to compete well against other approaches such as [9].

5.2 Face Localization

To study face localization, we performed experiments on the Faces in the Wild dataset [13]. The dataset contains faces in unconstrained environments. Our goal for this task is to localize the face in the image, as well as face parts such as eyes, nose, and mouth. We randomly select 200 images for training, and 100 images for testing. Although the dataset comes annotated with the identity of the persons in the image, it does not come with part annotations. We manually annotate all training and test images with bounding box information for the parts: Face, Left eye, Right eye, Nose, Mouth. Examples of the manual annotation are shown in Figure 4.

The face grammar has symbols Face (F), Left eye (L), Right eye (R), Nose (N), and Mouth (M). Each symbol has an associated set of poses of the form (x, y, s) , which represent a position and scale in the image. We refer to the collection of $\{L, R, N, M\}$ symbols as the parts of the face. The grammar has a single rule of the form $F \rightarrow \{L, R, N, M\}$. We express the geometric relationship between a face and each of its parts by a scale-dependent

¹The contour maps used in [9] may differ from ours since images in the BSDS500 have multiple annotations.

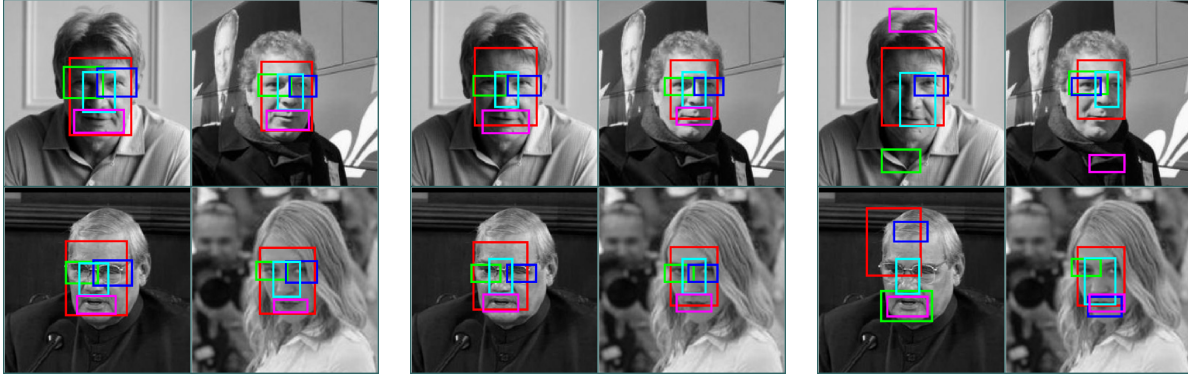


Figure 4: Localization results. Left: annotated ground-truth bounding boxes. Middle: results of the grammar model. Right: results of the baseline model using HOG filters alone. The parts are face (red), left eye (green), right eye (blue), nose (cyan), and mouth (magenta).

offset and region of uncertainty in pose space. The offset captures the mean location of a part relative to the face, and the region of uncertainty captures variability in the relative locations.

Concretely, suppose we had a Face with pose (x, y, s) . Then, for each part $Z \in \{L, R, N, M\}$, the Face would expand to a part Z somewhere in a uniform region centered at $(x', y') = (x, y) + sb_Z$. Having a part-dependent base offset b_Z allows us to express information such as “the mouth is typically near the bottom of the face” and “the nose is typically near the middle of the face”. The dependence of the offset on the scale s of the Face allows us to place parts in their correct position independent of the Face size. We model the relationship between scales of a Face and a part in a similar way. Modeling the relation between scales allows us to represent concepts such as large faces tending to have large parts. We learn the geometric parameters such as the part offsets by collecting statistics in the training data.

Figure 1 shows samples of scenes with one face generated by the grammar model we estimated from the training images in the face dataset. Note the location and scale of the objects varies significantly in different scenes, but the relative positions of the objects are fairly constrained. Samples of scenes with multiple faces are included in the supplemental material.

Our data model is based on HOG filters [4]. We train HOG filters using publicly-available code from [6]. We train separate filters for each symbol in the grammar using our annotated images to define positive examples. Our negative examples are taken from the PASCAL VOC 2012 dataset [5], with images containing the class “People” removed.

The score of a HOG filter is real-valued. We convert this score to a probability using Platt’s method [18], which involves fitting a sigmoid. This allows us to estimate $P(X(A, \omega) = 1 | \text{score})$ for each symbol $A \in \{F, L, R, N, M\}$. For the observation model we require a quantity that can be interpreted as $P(\text{score} | X(A, \omega) = 1)$, up to a proportionality constant. We note that $P(\text{score} | X(A, \omega) = 1) \propto P(X(A, \omega) = 1 | \text{score}) / P(X(A, \omega) = 1)$. We approximate $P(X(A, \omega) = 1)$ using the self-rooting probability, ϵ_A . To connect the data model to the grammar, the normalized scores of each filter are used to define messages into the corresponding bricks in the factor graph.

The result of inference with our grammar model leads to the (approximate) probability that there is an object of each type in each pose in the image. We show detection and localization results on images with multiple faces in the supplementary material. To quantify the performance of the model for localizing the face and its parts on images containing a single face we take the highest probability pose for each symbol. As a baseline we consider localizing each symbol using the HOG filter scores independently, without using a compositional rule.

Figure 4 shows some localization results. The results illustrate the context defined by the compositional rule is crucial for accurate localization of parts. The inability of the baseline model to localize a part implies the local image evidence is weak. By making use of contextual information in the form of a compositional rule we can perform accurate localization despite locally weak image evidence.

We provide a quantitative evaluation of the grammar model the baseline model in Table 1. The Face localization accuracy of both models are comparable. However, when attempting to localize smaller objects such as eyes, context

| Model | Face | Left Eye | Right Eye | Nose | Mouth | Average |
|---------------|-------------|-------------|-------------|------------|-------------|---------|
| HOG filters | 14.7 (18.7) | 33.8 (39.7) | 37.9 (35.1) | 8.9 (18.1) | 24.6 (35.0) | 24.0 |
| Grammar-Full | 13.1 (17.1) | 6.6 (12.4) | 8.2 (16.5) | 5.5 (10.6) | 11.4 (17.7) | 9.0 |
| Grammar-Parts | 13.8 (18.3) | 6.1 (10.8) | 8.8 (19.1) | 7.4 (15.1) | 12.1 (19.1) | 9.7 |

Table 1: Mean distance of each part to the ground truth location. Standard deviations are shown in brackets. Grammar-Full denotes the grammar model of faces with filters for all symbols. Grammar-Parts denotes the grammar model with no filter for the face symbol. The grammar models significantly outperform the baseline in localization accuracy. Further, the localization of the Face symbol for Grammar-Parts is very good, suggesting that context alone is sufficient to localize the face.

becomes important since the local image evidence is ambiguous. We also ran an experiment with the grammar model *without* a HOG filter for the face. Here, the grammar is unchanged but there is no data model associated with the Face symbol. As can be seen in the bottom row of Table 1, we can localize faces very well despite the lack of a face data model, suggesting that contextual information alone is enough for accurate face localization. Inference using the grammar model on a (250×250) test image took 2 minutes.

6 Conclusion

Probabilistic scene grammars define priors that capture relationships between objects in a scene. By using a factor graph representation we can apply belief propagation for approximate inference with these models. This leads to a robust algorithm for aggregating local evidence through contextual relationships. The framework is quite general and the practical feasibility of the approach was illustrated on two different applications. In both cases the contextual information provided by a scene grammar proves fundamental for good performance.

Acknowledgements

We would like to thank Stuart Geman and Jackson Loper for many helpful discussions about the topics of this research.

References

- [1] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, May 2011.
- [2] Elie Bienenstock, Stuart Geman, and Daniel Potter. Compositionality, MDL priors, and object recognition. In *Advances in Neural Information Processing Systems*, pages 838–844, 1997.
- [3] Michael Burl, Markus Weber, and Pietro Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *European Conference on Computer Vision*, pages 628–641. Springer, 1998.
- [4] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.
- [5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
- [6] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. <http://people.cs.uchicago.edu/~pff/latent-release4/>.
- [7] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

- [8] Pedro F. Felzenszwalb and David McAllester. Object detection grammars. *Univerity of Chicago Computer Science Technical Report 2010-02*, 2010.
- [9] Pedro F. Felzenszwalb and John G. Oberlin. Multiscale fields of patterns. In *Advances in Neural Information Processing Systems*, pages 82–90, 2014.
- [10] Sanja Fidler, Marko Boben, and Aleš Leonardis. Learning a hierarchical compositional shape vocabulary for multi-class object representation. In *ArXiv:1408.5516*, 2014.
- [11] Martin A. Fischler and Robert A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on computers*, (1):67–92, 1973.
- [12] Ross B. Girshick, Pedro F. Felzenszwalb, and David Mcallester. Object detection with grammar models. In *Advances in Neural Information Processing Systems*, pages 442–450, 2011.
- [13] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.
- [14] Ya Jin and Stuart Geman. Context and hierarchy in a probabilistic image model. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2145–2152, 2006.
- [15] Frank R Kschischang, Brendan J Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.
- [16] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999.
- [17] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence*, pages 467–475, 1999.
- [18] John C. Platt. Probabilities for SV machines. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.
- [19] Przemyslaw Prusinkiewicz and Aristid Lindenmayer. *The algorithmic beauty of plants*. Springer, 1991.
- [20] Zhuowen Tu, Xiangrong Chen, Alan L. Yuille, and Song-Chun Zhu. Image parsing: Unifying segmentation, detection, and recognition. *International Journal of computer vision*, 63(2):113–140, 2005.
- [21] Yibiao Zhao and Song-Chun Zhu. Image parsing with stochastic scene grammar. In *Advances in Neural Information Processing Systems*, pages 73–81, 2011.

Supplementary Material

A Contour Detection Results

Figures 5 and 6 show more contour detection results on images from the BSDS500 at a larger resolution so the reader can examine more details.

B Multiple Faces

In Figure 7 we show unconstrained samples from the grammar model for faces. Note how the model generates scenes with multiple faces, and also generates parts that appear on their own, since every symbol has a non-zero probability of self-rooting.

In Figure 8 we show localization results for images with multiple faces. In this case we show the top K poses for each symbol after performing non-maximum suppression, where K is the number of faces in the image.

In general we do not know in advance the number of symbols of each type that are present in the scene. In this case it is not possible to simply select the top K poses for each symbol. A different strategy is to use a threshold, and select bricks that have marginal probabilities above the threshold to generate detections. The particular threshold used depends on the desired trade-off between false positives and false negatives. A threshold for a particular application can be set by examining a Precision-Recall curve. For three example images, we manually selected a single threshold for detection that leads to good results. In Figure 9 we show *all* Face bricks with marginal probability above the threshold, after applying non-maximum suppression.

C Contextual Influence

Figure 10 shows the results of inference when conditioning on various parts being in the image in specific poses. The grammar used here was a simplified version of the face grammar in the main paper. The symbols are Face (F), Eye (E), Nose (N), and Mouth (M). The pose of each symbol is the set of pixels in the image (there is no scale variation). The only compositional rule is $F \rightarrow \{E, E, N, M\}$. Note that we use the same symbol to represent the left and right eyes.

As can be seen in Figure 10, when we condition on the presence of a Face at a particular position (first row), the model “expects” to see parts of the face in certain regions in the image. When we condition on the location of an Eye (second row), the model does not know whether the Eye should be a left eye or right eye, hence there are two modes for the location of the Face, and two modes for the location of another Eye. Intuitively LBP is performing the following chain of reasoning: (1) the eye that is known to be present can be a left or right eye, (2) there are two possible regions of the image in which the face can occur, depending on whether the eye that is known to be present is a left or right eye, and finally (3) given each possible pose for the face, the other parts of the face should be located in a particular spatial configuration. When we condition on more parts, LBP can infer that it is more likely for the face to be in one region of the image over another region, and the beliefs for the other face parts reflect this reasoning.

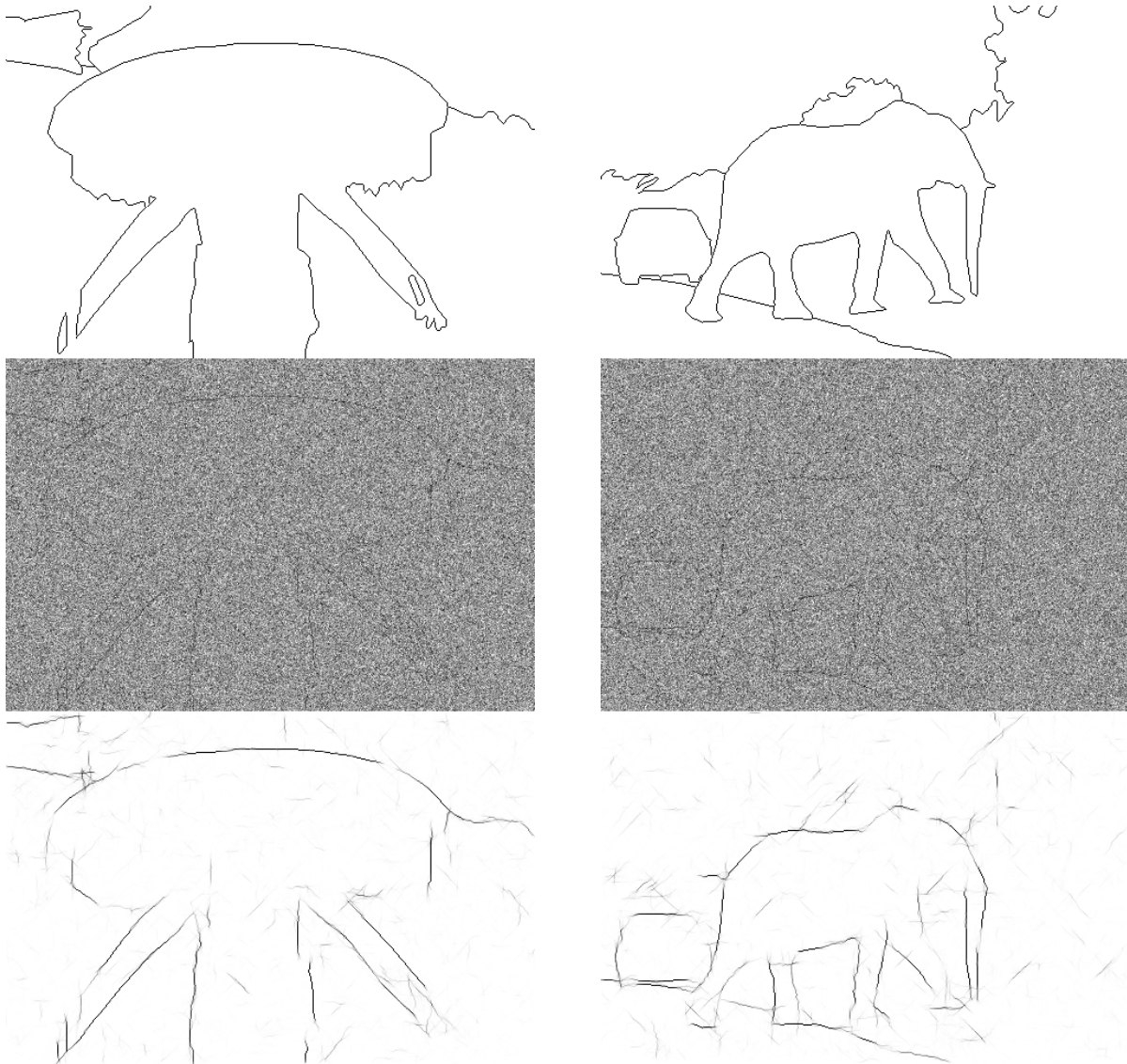


Figure 5: Contour detection results for images from the BSDS500 test set. Top: Ground-truth contour maps. Middle: Noisy observations. Bottom: The results of our model. The results show the estimated probability that a curve in the scene goes through each pixel, with dark values for high-probability pixels.

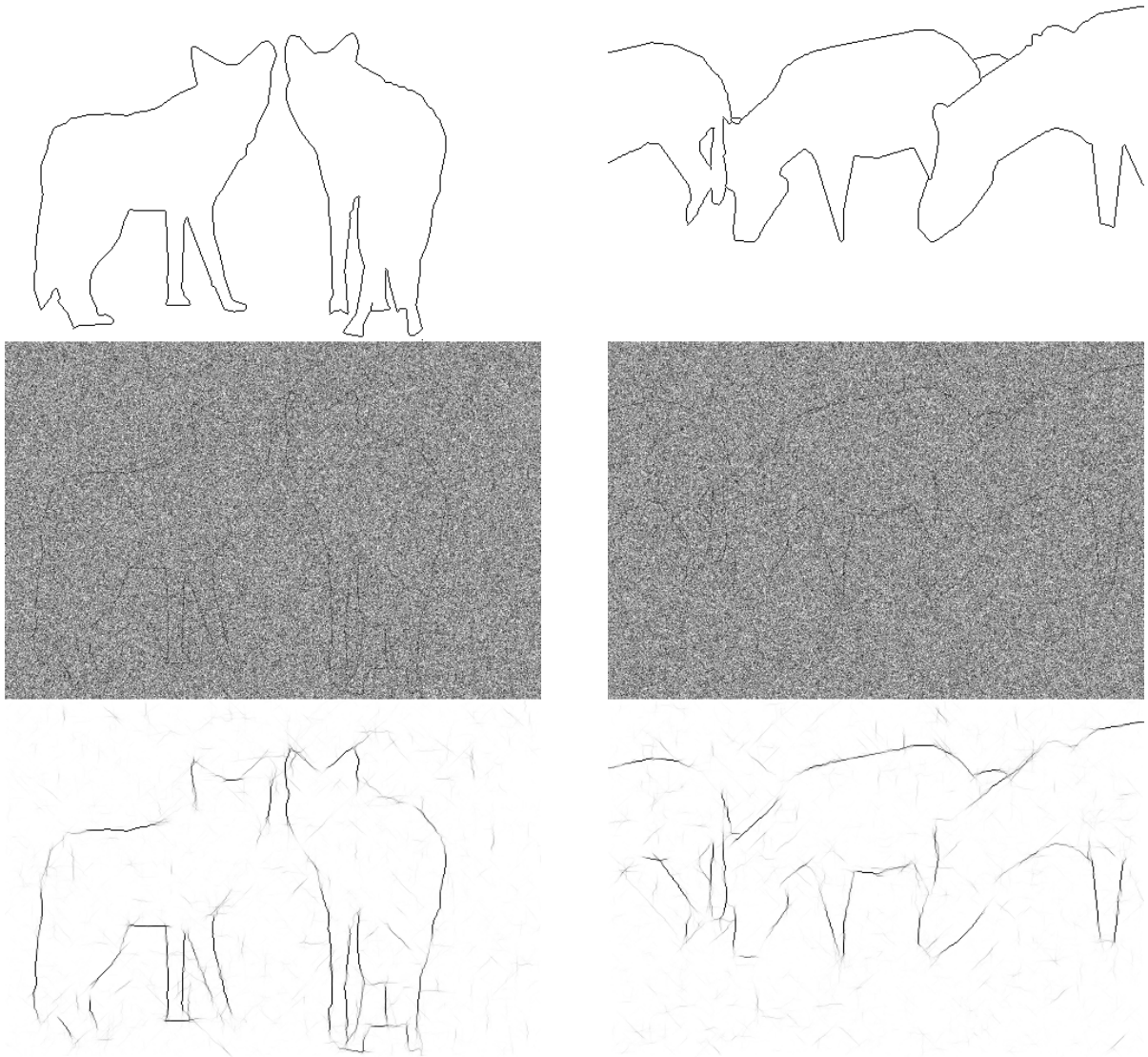


Figure 6: Contour detection results for images from the BSDS500 test set. Top: Ground-truth contour maps. Middle: Noisy observations. Bottom: The results of our model. The results show the estimated probability that a curve in the scene goes through each pixel, with dark values for high-probability pixels.

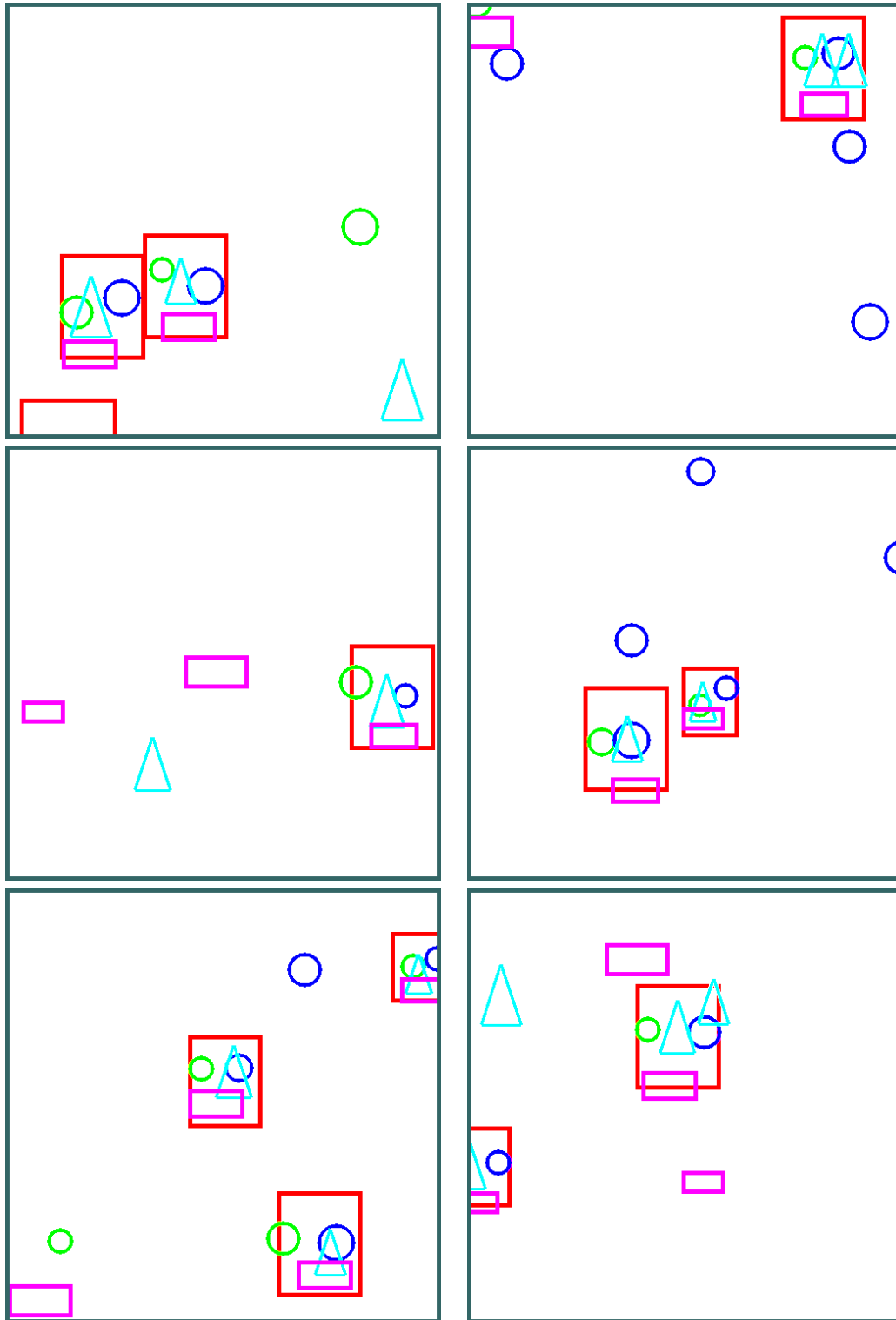


Figure 7: Unconstrained samples of scenes generated with the face grammar. The model generates scenes with multiple faces and parts can appear on their own since every symbol has a non-zero probability of self-rooting.

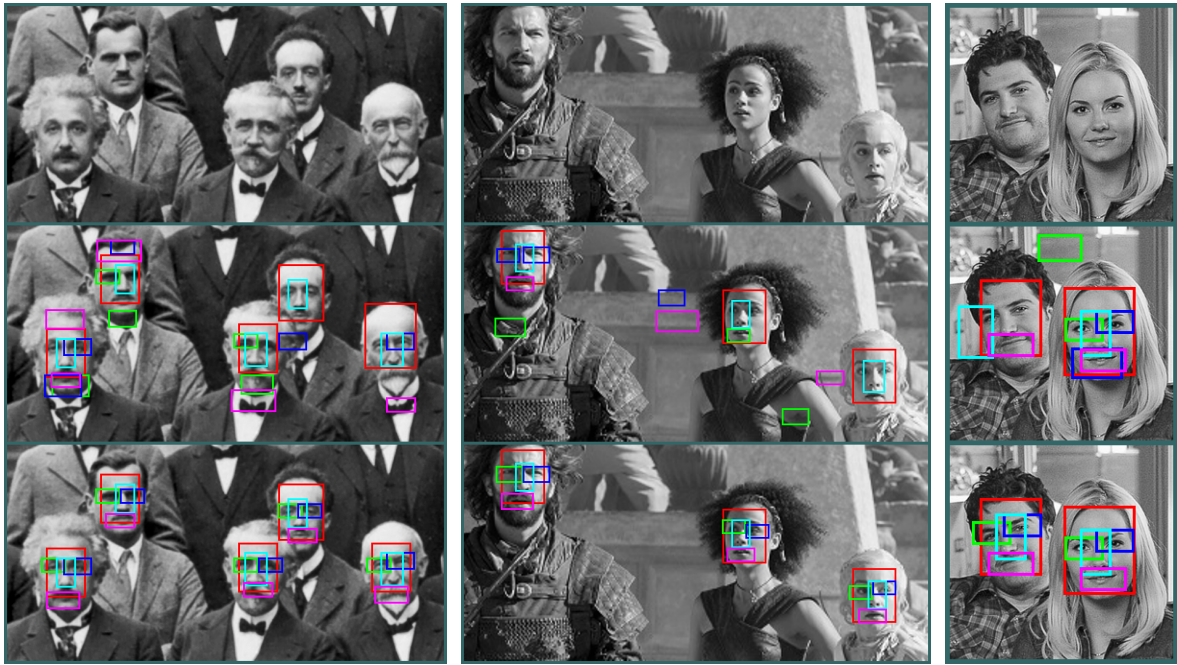


Figure 8: Localization results for images with multiple faces. Top: input images. Middle: localization with the HOG filter baseline. Bottom: localization with the full grammar model. We show the top K detections after non-maximum suppression, where K is the number of faces in the image. The parts are face (red), left eye (green), right eye (blue), nose (cyan), and mouth (magenta).

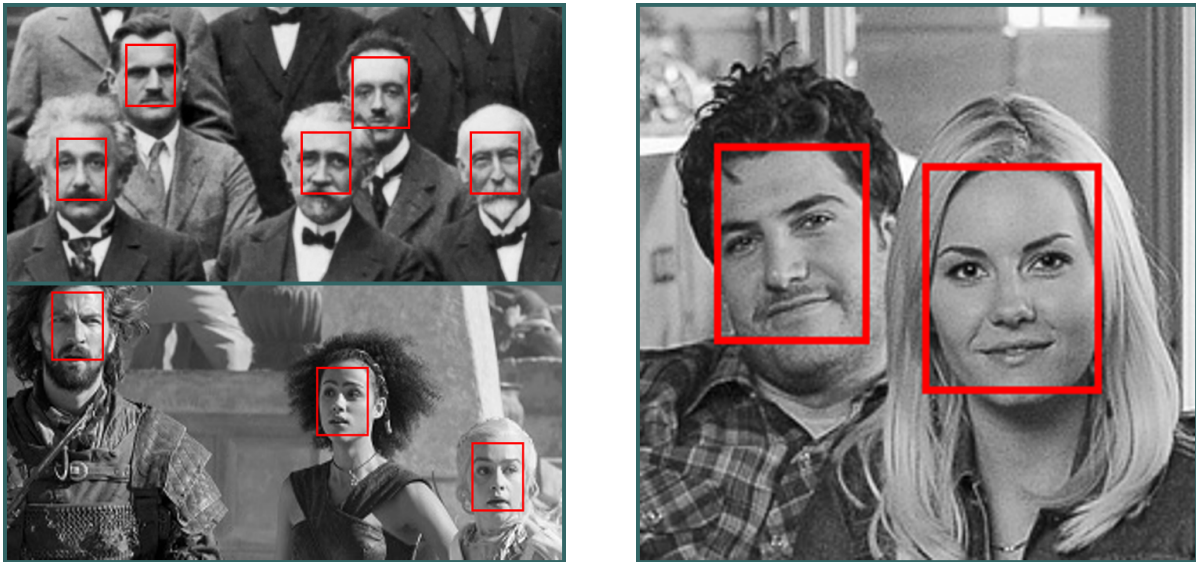


Figure 9: Face detection results using the full grammar model. We show *all* faces detected with a marginal probability greater than 0.25, after applying non-maximum suppression.

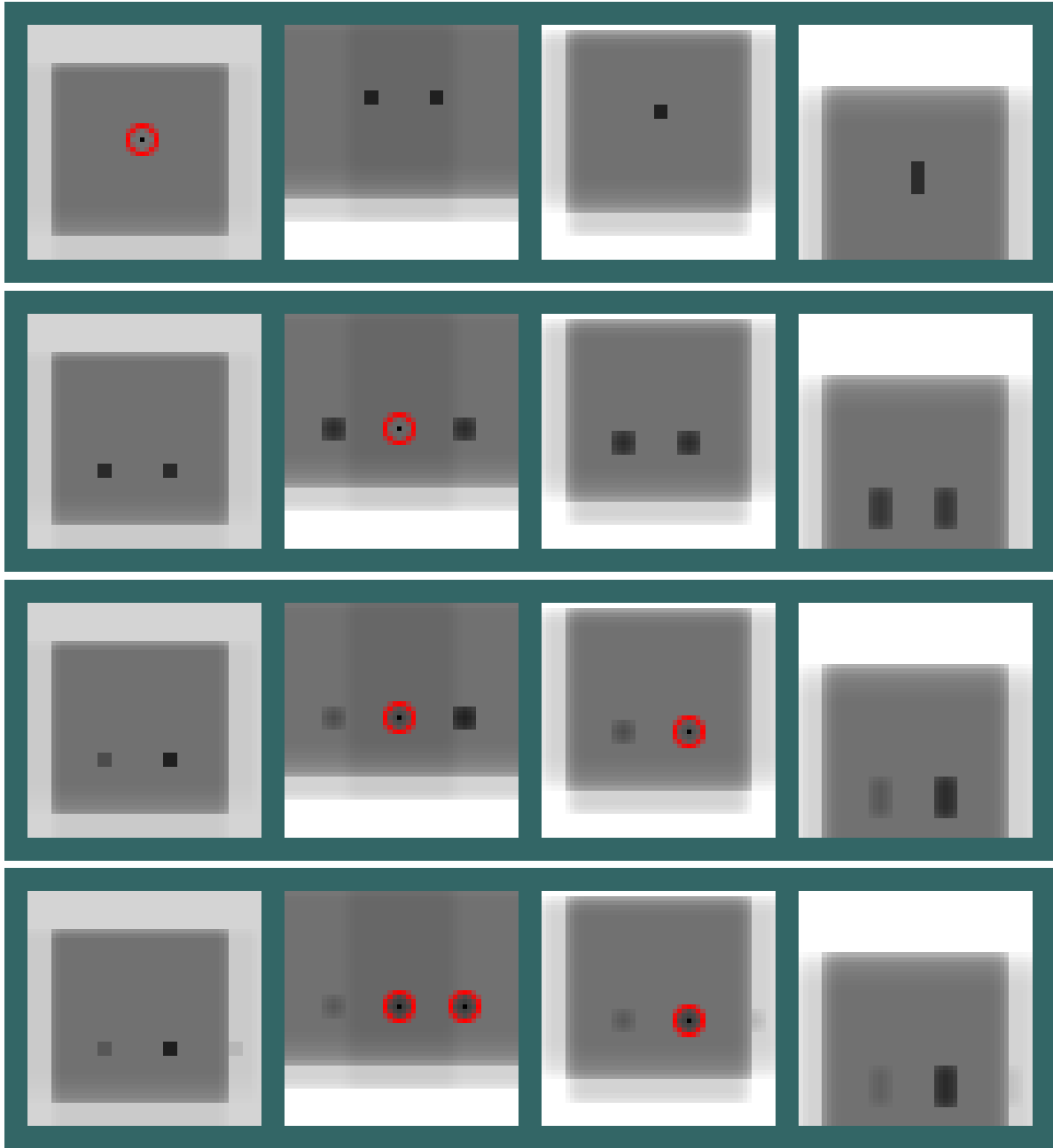


Figure 10: Contextual influence for a simple face grammar. The images show the log-marginal probabilities that different bricks are present in the scene, as computed by LBP. Darker pixels indicate higher probabilities. The symbols are (left to right): face, eye, nose, mouth. Each row shows the results of inference after conditioning on the presence of one or more bricks, indicated by red circles. As we condition on the presence of more parts, the contextual influence increases. For example, in row three, conditioning on the presence of a particular eye and nose brick make the presence of a face somewhere on the right side of the image highly likely.